IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Rahul SRIVASTAVA	Art Unit: 2175
Appl. No.: 10/711,791	Examiner: BASHORE, WILLIAM L
Date Filed: 10/05/2004	Atty. Docket No.: ORCL-006/OID-2004-061-01
For: Reducing Programming Complexity in Applications Interfacing With Parsers for Data elements Represented According to a Markup Languages	

Appeal Brief Under 37 CFR § 41.37

Mail Stop **Appeal Brief - Patents**
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Further to the Notice of Appeal filed on 7 October 2008, Appellants submit this appeal brief under 37 CFR § 41.37.

It is not believed that extensions of time or fees for net addition of claims are required, beyond those which may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, then such extensions of time are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required therefor (including fees for net addition of claims) are hereby authorized to be charged to Deposit Account No.: 20-0674.

As required by 37 C.F.R. § 41.37, this brief contains items under the following headings:

I. REAL PARTY IN INTEREST

II. RELATED APPEALS AND INTERFERENCES

III. STATUS OF CLAIMS

IV. STATUS OF AMENDMENTS

V. SUMMARY OF CLAIMED SUBJECT MATTER

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

VII. THE ARGUMENT

VIII. CLAIMS

IX. APPENDIX (Evidence and Related Proceedings)

I. REAL PARTY IN INTEREST

This application is assigned to Oracle International Corporation, by virtue of the assignment recorded on 10/05/2004 at Reel/Frame: 015218/0468 at the USPTO.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF CLAIMS

A. Pending Claims

Claims 1-49 are pending. Of these, claims 32-40 and 46 were not examined as being drawn to non-elected group.

Claims 1-31, 41-45 and 47-49 were examined. Of these, claims 1, 12, 21, 47 and 49 are independent claims. There are no cancelled claims.

B. Rejections

All the elected/examined claims 1-31, 41-45 and 47-49 were rejected.

C. Appealed Claims

All the rejected claims 1-31, 41-45 and 47-49 are subject of this appeal.

IV. STATUS OF AMENDMENTS

The Appellants have not filed any amendments after the Final Office Action dated 07/10/2008.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present invention is directed to the manner in which applications interface with parsers for data elements represented according to a markup language. Figure 1 of the subject application provides the corresponding context and is reproduced below. The specific support for the claimed elements is provided thereafter.

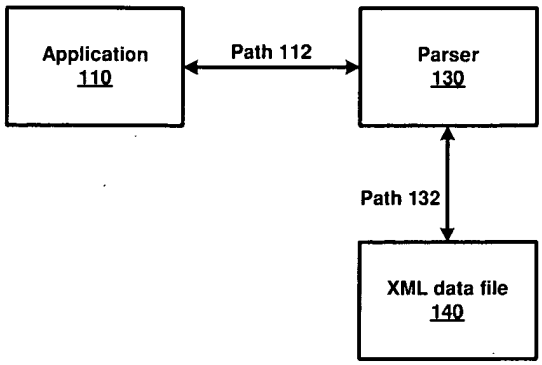

FIG. 1

Claims 1, 21 and 47 respectively relate to a method, computer readable medium and article of manufacture claims related to parsing a data file (e.g., XML data file 140 of Figure 1) containing data elements (Figure 4A) according to a markup language (paragraph 4 of the specification). The method of claim 1 expressly recites that the method is implemented in a parser 130. However, the features of the other claims are supported by the description associated with parser 130.

Parser 130 receives a file identifier of the data file to be parsed with an instruction to parse the data file 140 from an application 110 implemented external to the parser. This feature is further supported by step 210 of Figure 2 and paragraphs 42 and 43 of the specification. By providing the parser external to the application (paragraph 58 of the specification) as shown in Figure 1, several applications can share the same parser.

The parser retrieves a first data element (tag "Books" in line 402) from the specified data file (step 220 of Figure 2 and Paragraph 57 of the specification). The parser then determines a portion identifier as a relative location ("/Books" in row 422 Column 440 of Figure 4B, in general the XPATH of the data element according to XML

as noted in paragraph 12) for the first data element. The portion identifier indicates a relative location (last sentence of paragraph 57 of the specification) of the first data element with respect to another data element (paragraph 125) in the data file according to the markup language.

The parser provides the portion identifier and the first data element in association to the application, as shown with respect to step 240 of Figure 2 and described in paragraphs 46 and 47 of the specification. Paragraph 47 of the specification further discloses the manner in which the data element and the portion identifier are provided in association for event based parser techniques (XPath provided as a parameter while providing corresponding data element) and object based parsing technique (Xpath expression provided in the data structure representing XML data file 140 in a random access memory).

By having the parser provide such data element and the portion identifier in association, the programming complexity of applications interfacing with parsers can be reduced (paragraph 30 of the specification).

Independent claim 47 further recites that the data element is present stored in the data file prior to receiving the file identifier from the application. Such a feature is supported by Figure 4A and associated description in paragraph 56 of the specification, in combination with Figure 1 (XML data file 140).

Claim 3 recites that the parser provides the portion identifier and the corresponding data element according to an application programming interface (API). The API is used by applications to obtain the portion identifier and the data element. The recitations are supported by the API shown in Figures 8A-8F in case of event based push parser.

Claim 4 recites that the API is defined to provide the portion identifier and the first data element as respective parameters of a "single" procedure call. This feature is supported by each of rows 842 and 843 of Figure 8C in that each row shows a respective

procedure call with xpath and xpathVal as parameters. The corresponding description is provided in paragraph 96 of the specification.

Claim 5 recites that the parser is an event-based parser, which is defined in paragraphs 7-9 of the specification.

Claim 6 recites that the parsing is performed by an object oriented parser (paragraph 10 of the specification). The parser is recited as storing the first data element and the portion identifier in a data structure (DOM, Paragraphs 10, 67 and 68), with application obtaining the first data element (line 587 of Figure 5B) and the corresponding portion identifier (line 555 of Figure 5B) from the data structure. Paragraphs 70 and 71 provide the corresponding description.

Claim 10 recites that parsing is performed according to a push parsing approach (paragraph 8 of the specification), wherein said API provides a procedure for the parsing to provide the Xpath to the application. This feature is supported by each of rows 842 and 843 of Figure 8C, in that each row shows a respective procedure call with xpath as a parameter.

Independent claims 12 recites a method of implementing an application (e.g., 110 of Figure 1) using data contained in a data file 140. The data file is recited as containing multiple data elements (Figure 4A) according to a markup language (e.g., XML as described in paragraph 5).

The application is recited as instructing a parser to parse the data file of interest, with the parser being implemented external to the application (step 310 of Figure 3 and paragraph 51 of the specification). This feature is further supported by line 520 of Figure 5A and associated description in paragraph 66 in case of Object oriented parser, line 616 of Figure 6 and associated description in paragraph 79 in case of push based (SAX) event parser with corresponding extensions according to aspects in other claims, lines 917 and 919 of Figure 9 and associated description in paragraphs 128/129 in case of a push based

event parser API, and lines 708-710 of Figure 7 and the description in paragraph 87 of the specification.

The application is recited as obtaining in association both a portion identifier and a data element from the parser, with the portion identifier indicating a relative location of the data element with respect to another data element in the data file according to the markup language (step 325 of Figure 3 and paragraph 52 of the specification). This feature is supported by lines 555 (portion identifier) and 587 (data element) of Figure 5B and associated description in paragraph 71 in case of Object oriented parser, lines 621, 629 and 637 of Figure 6 and associated description in paragraph 80 in case of push based (SAX) event parser with corresponding extensions according to aspects in other claims, lines 921 and 927 of Figure 9 and associated description in paragraphs 130 in case of a push based event parser API, and lines 713, 716 and 719 of Figure 7 and the description in paragraph 88 of the specification.

The application then processes the obtained data element and the portion identifier (step 330 of Figure 3 and the description of Paragraph 53 of the specification). This feature is further supported by the last sentence of paragraphs 79 and 88, which indicates the processing can be any business logic as desired by a programmer of the application.

By receiving such data element and the portion identifier in association, the programming complexity of applications can be reduced (paragraph 30 of the specification).

Independent claim 49 recites an application (110 in Figure 1) and an event based parser (parser 130 in combination with paragraphs 7-9 of the specification). The application instructs the event based parser to parse a data file of interest (step 310 of Figure 3). In response, parser 130 retrieves data from the data file (step 230 of Figure 2), sets a portion identifier to a relative location ("/Books" in row 422 Column 440 of Figure 4B, in general the XPATH of the data element according to XML as noted in paragraph 12), and then provides to the application in association the data element and the portion identifier (step 240 of Figure 2) as a sequence of events without receiving a request

(paragraphs 7-9 of the specification; lines 621, 629 and 637 of Figure 6 and associated description in paragraph 80 in case of push based (SAX) event parser with corresponding extensions according to aspects in other claims, lines 921 and 927 of Figure 9 and associated description in paragraphs 130 in case of a push based event parser API).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claim 49 was rejected under 35 USC § 101 allegedly as being directed to non-statutory subject matter.

Claims 1-9,12-29, 41-45, and 47-49 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Nielsen (hereinafter Nielsen) PG Pub 2004/0205567, in view of Cseri *et al* (hereinafter Cseri), U.S. PG Pub. No. US 2003/0046317.

Claims 10-11 and 30-31 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Nielsen and Cseri, and further in view of Imamura ct al. (hereinafter Imamura), U.S. PG Pub. No. US 2004/0261019.

Appellants request the Board to review each of these rejections.

VII. THE ARGUMENT

VII. A. Rejection Under 35 U.S.C. § 103

Claim 49 has been rejected under 35 U.S.C. § 101 allegedly as being directed to non-statutory subject matter. The Examiner further asserts that the claim language does not include hardware.

It is first respectfully noted that the various terms "technological art" and "concrete, useful and tangible result", used by the Examiner, have been held to be no longer valid (page 23, line 18-22 of the published Judgment dated 30 October 2008) by CAFC *In Re Bernard L. Bilski and Rand A. Warsa* (Hereafter "Bilski").

It is further noted that much of the analysis there and the terms used by the Examiner are believed to be relevant to claims directed to "processes", and not a "digital

processing unit" (of claim 49), which falls under the machine category under 35 USC § 101.

However, Bilski further expressly states, "The raw ***materials*** of many information age processes, however, are electronic signals and electronically manipulated data." (Page 25 lines 12-13, *ibid, **Emphasis Added***).

Appellants respectfully contend that the claimed application and event based parsers are such 'materials', which have been designed a particular way, as claimed.

Furthermore, the claim is not an "abstract idea" as alleged by the Examiner since it does not pre-empt the use of any fundamental principles. The claim is directed to an improvement in a specific context, while not covering what is not claimed in that context. Thus, the fundamental concern of pre-empting all possible uses of an idea, is not present as relevant to claim 49.

The Board is accordingly requested to overturn the rejection. The Board is further respectfully requested to address the rejection of claim 49 under 35 U.S.C. § 103, irrespective of decision on the rejection based on 35 U.S.C. § 101.

VII. Rejections Under 35 U.S.C. § 103

The Examiner has clearly not met the burden of setting forth a prima facie case of obviousness. MPEP § 2143 sets forth the basic requirements for establishing a prima facie case of obviousness under 35 USC § 103 (a).

Applicable Law

To establish a *prima facie* case of obviousness, three basic legal criteria must be met.

(L1) First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings;

(L2) Second, there must be a reasonable expectation of success; and

(L3) Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations.

Furthermore, (L4) the Office is permitted to give each term in the claim its broadest reasonable construction **consistent with the specification**. Phillips v. AWH Corp., 415 F.3d 1303, 75 USPQ2d 1321 (Fed. Cir. 2005). The PTO determines the scope of claims in patent applications not solely on the basis of the claim language, but upon giving claims their broadest reasonable construction **"in light of the specification as it would be interpreted by one of ordinary skill in the art."** In re Am. Acad. of Sci. Tech. Ctr., 367 F.3d 1359, 1364[, 70 USPQ2d 1827] (Fed. Cir. 2004).

Introduction to the References

Nielsen is the primary reference relied upon by the Examiner. Nielsen relates to imbedding XML fragments in (the content specified in) XML documents during run-time. In particular, the solutions of Nielsen are applicable in testing of web services in which XML documents would have content that is not available at the time of creation (of the XML document), but would be available later during run time.

In particular, Figure 1 of Nielsen provides the general context in which testing is performed. Test suite file 110 specifies interactions/tests in test suite file. Some of the tests generate "actual responses", which are used by later tests specified in the test suite file 100 (paragraphs 48-50 of Nielsen).

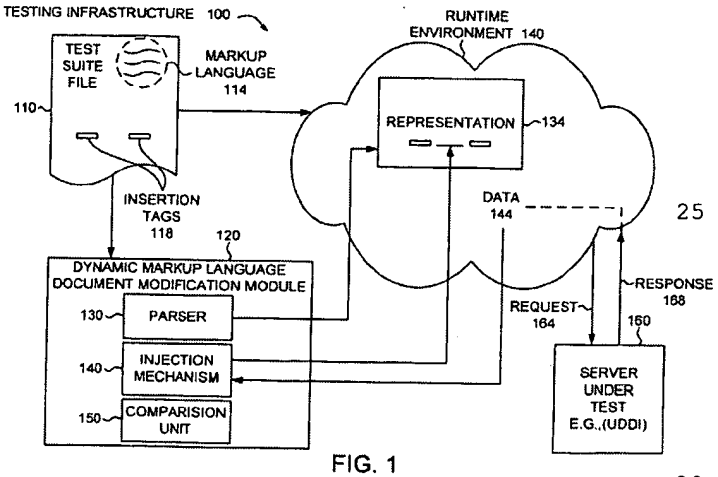

FIG. 1

Thus, test suite file 110 is parsed by parser 130 into representation 134 (paragraph 36 of Nielsen). Injection mechanism 140 performs the interactions/tests on server under test 160, which provides actual responses. The actual responses are inserted (by addition of nodes) into

representation 134 according to the "reference flags" in test suite file 110. The inserted information is used to perform later tests(paragraphs 49-50 of Nielsen).

Figure 2 of Nielsen provides the further details of dynamic markup language document modification module 120. In particular, there is shown a parser 130 which accepts markup language document 110 as input and forms representation 134 in runtime environment 140. The representation is further described as being in the form of document object model (DOM) (paragraph 36 of Nielsen).

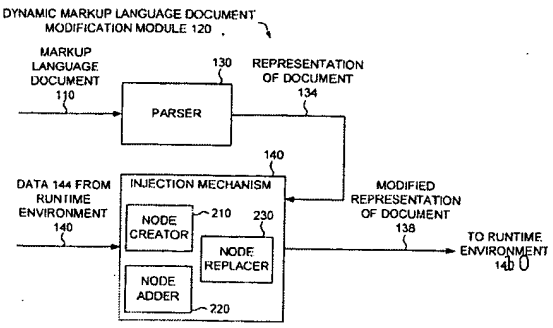

FIG. 2

The flowchart of Figure 3 illustrates the manner in which interactions in the test suite file are processed. The flowchart of Figure 4 illustrates the manner in which references are replaced.

The references are replaced in the DOM representation (paragraph 50 of Nielsen) according to a pre-specified convention described in paragraphs 63 and 64 of Nielsen.

The operation of the flowcharts of Figures 3 and 4 is illustrated below with the remaining figures of Nielsen.

Figure 5 of Nielsen depicts a test suite file in the form of an XML document. The test suite is shown containing two interactions. Each interaction corresponds to one test. In the example there, the "actual response" or result of the first interaction/test is used a part of the input to the second test/interaction. Since the result would be created dynamically (later), the invention provides enabling mechanisms as described below.

The enabling mechanism is illustrated with respect to two reference flags (shown in **bold** in Nielsen) in the second interaction of Figure 5. The two reference flags are identified below:

1. <ref:key>../../../../interaction[1]/ actualResponse/result/theObject/@key </ref:key>;

2. <theObject ref:key = "../../../../interaction[1]/actualResponse/result /theObject /@key">... </theObject>

As described below, the first reference flag points to a specific location at which the first interaction would write its actual response (according to the design of DDMM 120, the application which performs the tests). The second reference flag also points to the same location.

However, in case of the first reference flag, a new node with the value of the attribute "key" is copied to the target (i.e., the location at which the first reference flag is present). In the case of the second reference flag, only the value pointed to, is copied to the target (paragraphs 63-64 of Nielsen).

The first copying of above is performed to provide the appropriate inputs for the second interaction (step 310 of Figure 3 of Nielsen). The second copying is performed to check whether the actual response of the second interaction is equal to the expected response of the second interaction (step 360 of Figure 3 of Nielsen).

The above described concepts are now illustrated with respect to Figures 6-9 of Nielsen.

Figure 6 depicts the XML content after the actual response (shown in **bold**) for the first interaction is dynamically generated and added (paragraphs 80 of Nielsen). As noted above, this actual response is written to a pre-specified location and the reference flags are specified consistent with that convention.

Figure 7 depicts the XML content after the first reference flag is replaced by copying the value of the attribute key pointed to by the first reference flag (paragraphs 88 and 89 of Nielsen). A new node (<key>) with the copied value is created because the

source is a node/element (by virtue of "<ref:key>") and the target data is an attribute (by virtue of "@key"). The replaced text is again shown in **bold**.

Figure 8 depicts the XML content after the actual response (shown in **bold**) for the second interaction is dynamically generated and added (paragraph 97 of Nielsen). Again, the location at which it is added is according to pre-specified convention (consistent with the implementation of DDMM 120).

Figure 9 depicts the XML content after the second reference flag is replaced by copying only the value (12345), which is present at the location pointed to by the second reference flag (paragraph 99 of Nielsen). Only the value is copied since the source is an attribute (by virtue of "ref:key" being part of tag "<theObject>") and the target is also an attribute (same type). The replaced text is again shown in **bold**.

Cseri relates to providing an XML binary format when transmitting/receiving tag based descriptions (e.g., XML) of data. The binary formatting is achieved by tokenizing the tag and attribute names into variable sized numeric tokens, thereby obviating the need for repetitive or redundant storage of lengthy Unicode words, etc. The binary formatting minimizes parsing time and the generation of overhead incident to the formatting and parsing of data. Parsing time is thereby substantially decreased and generally, the size of the resulting file decreases too (See Abstract of Cseri).

Basis for Appellant's Positions

There are several reasons, based on which Appellants contend that presented claims are allowable over the art of record.

As a threshold matter, it is first noted that Nielsen is directed to a problem that is unrelated to the problem solved by the present invention. As noted above, the subject patent application is directed to a novel parser, which allows novel applications to be implemented. On the other hand, Nielsen is directed to a imbedding XML fragments "in the content" specified by XML documents (even though the title and higher level

description merely states that the fragments are imbedded in XML documents during run-time).

The general high level differences manifest in the claims being distinguishable on several grounds from Nielsen and the art of record.

For example, claim 1 recites that the method is implemented within a parser. As explained above, Nielsen also discloses a parser 130. As may be readily observed from Figures 1 and 2 of Nielsen, parser 130 receives markup language document 110 and form representation of document 134 in runtime environment 140.

The Examiner ignores the boundaries of parser 140 and attempts to map the various claimed features on operations external to parser 130 (in particular injection mechanism 140) onto various features of claim 1.

Without the hindsight gleaned from the Appellant's disclosure, one skilled in the relevant arts would not equate the claimed parser to the combination of parser and injection mechanism 140 of Figure 2. Alternatively, without such hindsight, one skilled in the relevant arts would not integrate the features described in injection mechanism 140 into parser 130.

For that reason alone, independent claim 1 is allowable over the art of record.

As another basis, Appellants note that construction of independent claim 1 requires that a parser receive a file identifier from an application, and the combination of data element and corresponding portion identifier be provided (in association) to the **same application by the parser**.

It is unclear to the Appellant how the Examiner finds this feature in the disclosure of Nielsen. The record does not make it clear which portions of Nielsen the Examiner equates to the claimed application and which portions to the claimed parser.

The Appellants had pointed out this difference in lines 1-3 page 11 of the amendment dated 28 May 2007. As related to this aspect, the Examiner asserts:

> ... Nielsen discloses the above analysis and replacement methods conducted during runtime (using an application) (Nielsen Abstract), ***therefore data (i.e. portion identifiers, data elements, etc.) are provided accordingly.*** (Lines 20-21 Page 3 of the Final Office Action Dated 7/10/2008, ***Emphasis Added***)
>
> It is additionally noted that it is well within reason to interpret Nielson's invention in the ***form of an executable set of instructions (i.e. an application),*** for carrying out its invention. (Lines 4-5 Page 6 of the Final Office Action Dated 28 March 2007, ***Emphasis Added***)

From the above, Appellants understand the Examiner's position to be mere presence of the fragments of the claim in the references is sufficient, without regard to where such fragments are operative in the environment of the reference.

To the contrary, Appellants note that a skilled practitioner, by reading the disclosure of the subject application, would readily understand that each of the claimed application and the claimed parser inherently have independent existence (e.g., each could operate in its own allocated memory space or one can be executed without executing the other, etc.).

Accordingly, the Examiner would be required to show similar independent blocks in the art of record in mapping the claimed application and claimed parser.

The Examiner simply ignores such a requirement and continues to maintain that Nielsen alone shows the claimed application and the claimed parser (except admitting that Nielsen does not teach that the parser is external to the application).

The Examiner's position would thus have at least several legal deficiencies. For example, Nielsen and Cseri together would not teach several features of claim 1 (as would be interpreted by one skilled in the relevant arts). In addition, the assertions of the

Examiner would only be based on impermissible hindsight gleaned from Appellant's disclosure.

Claim 1 would thus be allowable for these deficiencies as well.

As yet another basis, Appellants note that claim 1 requires that the first data element be **retrieved from the data file**, and such a data element be provided in association with the portion identifier to the application. The Examiner equates the claimed portion identifier to the XPath referred to in paragraphs 52 and 62 of Nielsen (see page 3 lines 13-14 of the Final Office Action dated 07/10/2008). The two paragraphs are reproduced below:

> [0052] For each interaction x in interactions, the following steps are performed. In step 310, the references in the current interaction's request are replaced with the thing the references point to. This step is described in greater detail hereinafter with reference to FIG. 4. A technology, such as Xpath, may be utilized for this purpose. One aspect of the present invention is the use of a referencing technology, such as Xpath for referencing a portion of the same document (i.e., a portion internal to the document). In the prior art, the referencing technology has been generally limited to referencing a portion of a document external to the document where the reference exists. For example, an Xpath reference in a first document is typically utilized to print a portion of a second document.
>
> [0062] When the node's name begins with "ref:" (this node is hereinafter referred to as the source), the following steps are performed. First, the replacement module finds what the attribute or element's XPath points to (which is referred to herein as a target). Second, the replacement module removes the source node and replaces the source node with a new node of the same type (e.g., attribute or element). The name for the node is the name of the source node with the "ref:" portion removed.

Unfortunately, the XPATH in the two paragraphs above does **not point to a data element retrieved from the data file**.

Rather the XPATH points to the location where the data 144 received from server under test 160 would be stored. The Examiner's attention was drawn to this aspect in the response dated 28 May 2007 (page 10 lines 15-28).

For this reason also, claim 1 is allowable over the art of record.

Claim 5 is independently allowable over the art of record in reciting that the parser is an event based parser. The Examiner admits that Nielsen does not teach an event based parser, but instead relies on the Cseri and/or Imamura for such a feature. The combination would clearly be based on impermissible hindsight at least for the below reason.

The Board's attention is directed to the fact that Nielsen expressly relies on the DOM structure (representation134) generated by parser 130 for the fundamental purpose of imbedding. An event based parser would not generate the representation 134, which is relied by injection mechanism 140 for the fundamental purpose for which Nielsen is designed.

Accordingly, the modification proposed by the Examiner (based on Cseri) would render the embodiments of Nielsen inoperative. This would clearly point to impermissible hindsight exercised by the Examiner in rejecting claim 4 based on the combined teachings of Nielsen and Cseri.

Independent claim 12 is also allowable over the art of record (for the reasons noted above) in reciting that the method is being implemented in an application, which is implemented external to the parser, and also that the application receives a portion identifier and data element in association.

Independent claim 21 parallels claim 1 in several respects (without reciting that the parser is implemented external to the application) and is allowable for some of the reasons noted above.

With respect to independent claim 47, the Appellants had stated the following, but have not received a direct response from the Examiner in the Final Office Action dated 07/10/2008:

> Currently amended independent claim 47 is also allowable over Nielsen in reciting that the data element is present stored in the data file prior to the receiving of the file identifier of the data file.

Again, it is unclear from the record to which specific teaching of Nielsen the Examiner is equating the claimed data element.

However, to advance prosecution, it is pointed out that data 144 is generated dynamically by server 160 after parser 130 receives test suite file 110.

Therefore, any operation based on data 144 in Nielsen is believed not to anticipate the invention of claim 47. Thus, claim 47 is allowable over Nielsen at least for the reason noted above. (lines 7-16 of page 18 of Amendment Dated 04 April 2008)

VIII. CLAIMS

Claim 1 (Previously Presented): A method of parsing a data file containing a plurality of data elements according to a markup language, said method being implemented in a parser, said method comprising:

receiving a file identifier of said data file with an instruction to parse said data file from an application implemented external to said parser;

retrieving a first data element from said data file, wherein said first data element is comprised in said plurality of data elements;

determining a portion identifier of said first data element, wherein said portion identifier indicates a relative location of said first data element with respect to another data element in said data file according to said markup language; and

providing in association said portion identifier and said first data element to said application.

Claim 2 (Original): The method of claim 1, wherein said markup language comprises XML and said portion identifier comprises an XPath.

Claim 3 (Original): The method of claim 1, wherein said providing provides said portion identifier and said first data element according to an application programming interface (API), wherein said application obtains said portion identifier and said first data element by using one or more procedure calls according to said API.

Claim 4 (Original): The method of claim 3, wherein said API is defined to provide said portion identifier and said first data element as respective parameters of a single procedure call.

Claim 5 (Previously Presented): The method of claim 4, wherein said parser is an event-based parser.

Claim 6 (Original): The method of claim 3, wherein said parsing is performed by an object oriented parser, wherein said providing further comprises:

storing said first data element and said portion identifier in a data structure, wherein said application obtains said first data element and said portion identifier from said data structure.

Claim 7 (Original): The method of claim 3, wherein said API provides a procedure for said application to request that said XPath is to be provided either in abbreviated or non-abbreviated format.

Claim 8 (Original): The method of claim 3, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as XPaths.

Claim 9 (Original): The method of claim 3, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as values associated with said first data element.

Claim 10 (Original): The method of claim 3, wherein said parsing is performed according to a push parsing approach, wherein said API provides a procedure for said parsing to provide said Xpath to said application.

Claim 11 (Original): The method of claim 10, wherein said API allows said application to set a variable to a value to cause said procedure to provide attributes of said data element along with said XPath.

Claim 12 (Previously Presented): A method of implementing an application using data contained in a data file, said data file containing a plurality of data elements according to a markup language, said method being implemented in an application, said method comprising:

instructing a parser to parse said data file of interest, said parser being implemented external to said application;

obtaining in association both a portion identifier and a data element from said parser, wherein said portion identifier indicates a relative location of said data element

with respect to another data element in said data file according to said markup language, said data element being contained in said plurality of data elements; and

processing said data element and said portion identifier.

Claim 13 (Original): The method of claim 12, wherein said markup language comprises XML and said portion identifier comprises XPath.

Claim 14 (Original): The method of claim 12, wherein said obtaining obtains said portion identifier and said first data element according to an application programming interface (API).

Claim 15 (Original): The method of claim 14, wherein said API is defined to provide said portion identifier and said first data element as respective parameters of a single procedure call.

Claim 16 (Previously Presented): The method of claim 12, wherein said parser comprises an event-based parser.

Claim 17 (Original): The method of claim 14, wherein said parser comprises an object oriented parser, wherein said obtaining further comprises accessing said first data element and said portion identifier in a data structure.

Claim 18 (Original): The method of claim 14, wherein said API provides a procedure for said application to request that said XPath is to be provided either in abbreviated or non-abbreviated format.

Claim 19 (Original): The method of claim 14, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as XPaths.

Claim 20 (Original): The method of claim 14, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as values associated with said first data element.

Claim 21 (Previously Presented): A computer readable medium carrying one or more sequences of instructions causing a digital processing system to parse a data file containing a plurality of data elements according to a markup language, wherein execution of said one or more sequences of instructions by one or more processors contained in said digital processing system causes said one or more processors to perform the actions of:

receiving a file identifier of said data file from an application;

retrieving a first data element from said data file, wherein said first data element is comprised in said plurality of data elements;

setting a portion identifier of said first data element to equal a relative location of said first data element with respect to another data element in said data file according to said markup language; and

providing in association said portion identifier and said first data element to said application.

Claim 22 (Original): The computer readable medium of claim 21, wherein said markup language comprises XML and said portion identifier comprises an XPath.

Claim 23 (Original): The computer readable medium of claim 21, wherein said providing provides said portion identifier and said first data element according to an application programming interface (API), wherein said application obtains said portion identifier and said first data element by using one or more procedure calls according to said API.

Claim 24 (Original): The computer readable medium of claim 23, wherein said API is defined to provide said portion identifier and said first data element as respective parameters of a single procedure call.

Claim 25 (Original): The computer readable medium of claim 24, wherein said parsing is performed by an event-based parser.

Claim 26 (Original): The computer readable medium of claim 23, wherein said parsing is performed by an object oriented parser, wherein said providing further comprises:

storing said first data element and said portion identifier in a data structure, wherein said application obtains said first data element and said portion identifier from said data structure.

Claim 27 (Original): The computer readable medium of claim 23, wherein said API provides a procedure for said application to request that said XPath is to be provided either in abbreviated or non-abbreviated format.

Claim 28 (Original): The computer readable medium of claim 23, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as XPaths.

Claim 29 (Original): The computer readable medium of claim 23, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as values associated with said first data element.

Claim 30 (Original): The computer readable medium of claim 23, wherein said parsing is performed according to a push parsing approach, wherein said API provides a procedure for said parsing to provide said Xpath to said application.

Claim 31 (Original): The computer readable medium of claim 30, wherein said API allows said application to set a variable to a value to cause said procedure to provide attributes of said data element along with said XPath.

Claim 32 (Withdrawn): A computer readable medium carrying one or more sequences of instructions causing a digital processing system to implement an

application using data contained in a data file, said data file containing data according to a markup language, wherein execution of said one or more sequences of instructions by one or more processors contained in said digital processing system causes said one or more processors to perform the actions of:

instructing a parser to parse said data file of interest;

obtaining in association both a portion identifier and a data element from said parser, wherein said portion identifier identifies said data element in said data file according to said markup language, said data element being contained in said plurality of data elements; and

processing said data element and said portion identifier.

Claim 33 (Withdrawn): The computer readable medium of claim 32, wherein said markup language comprises XML and said portion identifier comprises XPath.

Claim 34 (Withdrawn): The computer readable medium of claim 32, wherein said obtaining obtains said portion identifier and said first data element according to an application programming interface (API).

Claim 35 (Withdrawn): The computer readable medium of claim 34, wherein said API is defined to provide said portion identifier and said first data element as respective parameters of a single procedure call.

Claim 36 (Withdrawn): The computer readable medium of claim 32, wherein said parser comprises an event-based parser.

Claim 37 (Withdrawn): The computer readable medium of claim 34, wherein said parser comprises an object oriented parser, wherein said obtaining further comprises accessing said first data element and said portion identifier in a data structure.

Claim 38 (Withdrawn): The computer readable medium of claim 34, wherein said API provides a procedure for said application to request that said XPath is to be provided either in abbreviated or non-abbreviated format.

Claim 39 (Withdrawn): The computer readable medium of claim 34, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as XPaths.

Claim 40 (Withdrawn): The computer readable medium of claim 34, wherein said API provides a procedure for said application to request that attributes related to said first data element are to be returned as values associated with said first data element.

Claim 41 (Previously Presented): method of claim 1, wherein said retrieving retrieves successive data elements from said data file, wherein said determining determines the portion identifier for all of the data elements in said data file, wherein each portion identifier is determined based on a hierarchy in which the data element is present according to said markup language, and wherein the portion identifier is determined upon retrieval of the corresponding data element from said data file.

Claim 42 (Previously Presented): The method of claim 41, wherein said providing provides each data element and corresponding portion identifier to said application before said retrieving retrieves a next data element from said data file.

Claim 43 (Previously Presented): The method of claim 12, wherein said obtaining obtains each of said plurality of data elements associated with the corresponding portion identifier from said parser.

Claim 44 (Previously Presented): The computer readable medium of claim 21, wherein said retrieving retrieves successive data elements from said data file, wherein said determining determines the portion identifier for each of said data elements in said data file upon retrieval of the data element based on a hierarchy in which the data element is present according to said markup language.

Claim 45 (Previously Presented): The computer readable medium of claim 44, wherein said providing provides each data element and corresponding portion identifier to said application before said retrieving retrieves a next data element from said data file.

Claim 46 (Withdrawn): The computer readable medium of claim 32, wherein said obtaining obtains each of said plurality of data elements associated with the corresponding portion identifier from said parser.

Claim 47 (Previously Presented): An article of manufacture for parsing a data file containing a plurality of data elements according to a markup language, said article of manufacture comprising:

means for receiving a file identifier of said data file from an application;

means for retrieving a first data element from said data file, wherein said first data element is comprised in said plurality of data elements and is present stored in said data file prior to said receiving;

means for determining a portion identifier of said first data element, wherein said portion identifier indicates a relative location of said first data element with respect to another data element in said data file according to said markup language; and

means for providing in association said portion identifier and said first data element to said application.

Claim 48 (Previously Presented): The method of claim 1, wherein said relative location comprises a hierarchical path from a root data element contained in said plurality of data elements, wherein said root data element occurs at the beginning of said data file.

Claim 49 (Previously Presented): A digital processing system comprising:

an application and an event based parser, wherein said parser is implemented external to said application such that multiple applications can use said parser,

wherein said application is designed to instruct said event based parser to parse a data file of interest, said data file contains a plurality of data elements,

in response, said parser designed to retrieve each of said plurality of data elements from said data file,

said parser to set a corresponding portion identifier of each of said plurality of data elements to equal a relative location of the data element with respect to another data element in said data file according to said markup language,

said parser to provide in association each of said plurality of data elements and corresponding portion identifier to said application as a corresponding one of a sequence of events without receiving a request for next data element from said application.

IX. APPENDIX

A. EVIDENCE APPENDIX: None

B. RELATED PROCEEDINGS APPENDIX: None